Unidad 8

BIG DATA - NoSQL

Prof. Mg. Ing. Roxana Martínez

UAI
Universidad Abierta Interamericana

UAIOnline Ultra

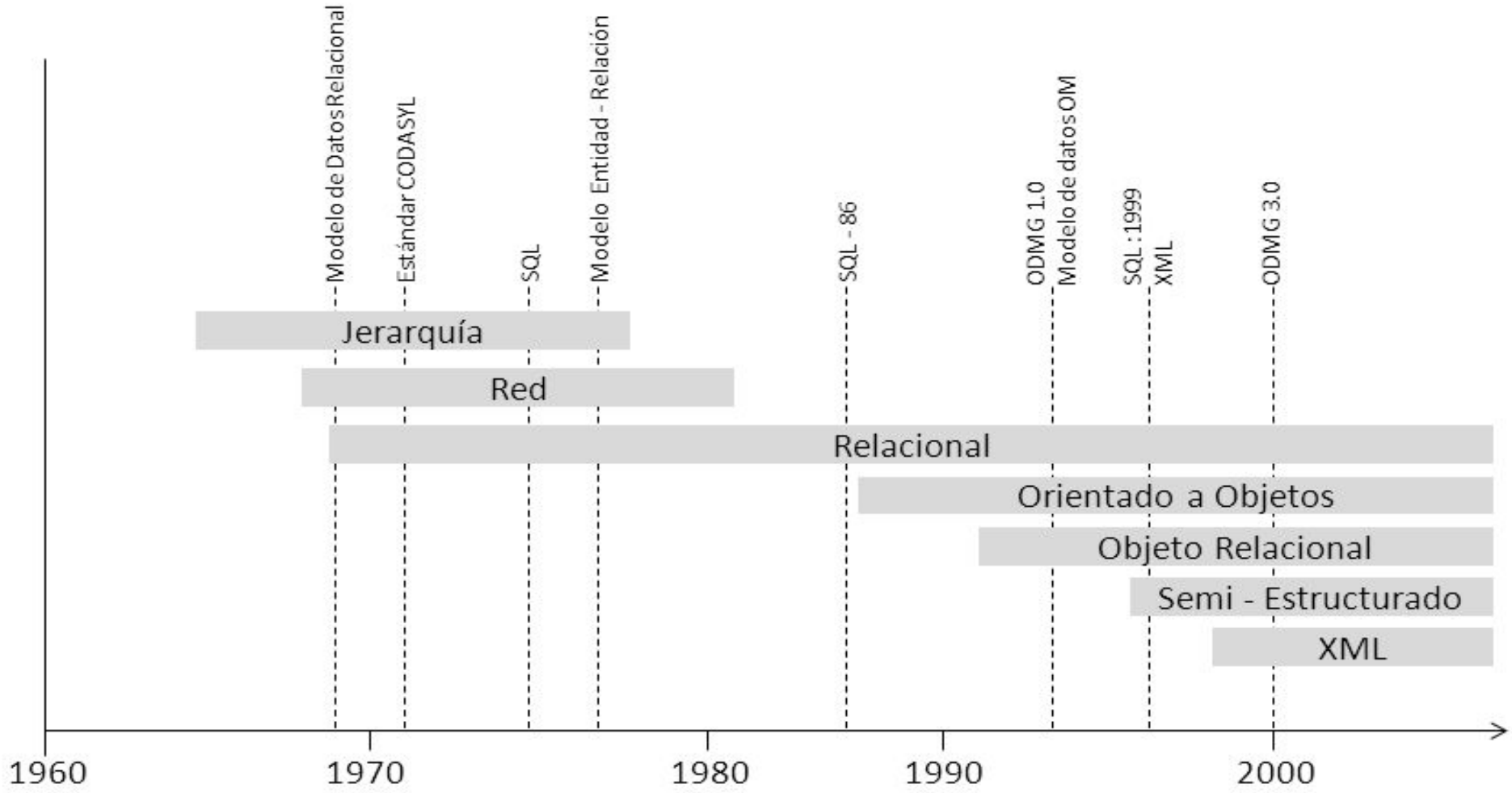
Big Data - NoSQL

Unidad 8

OBJETIVOS

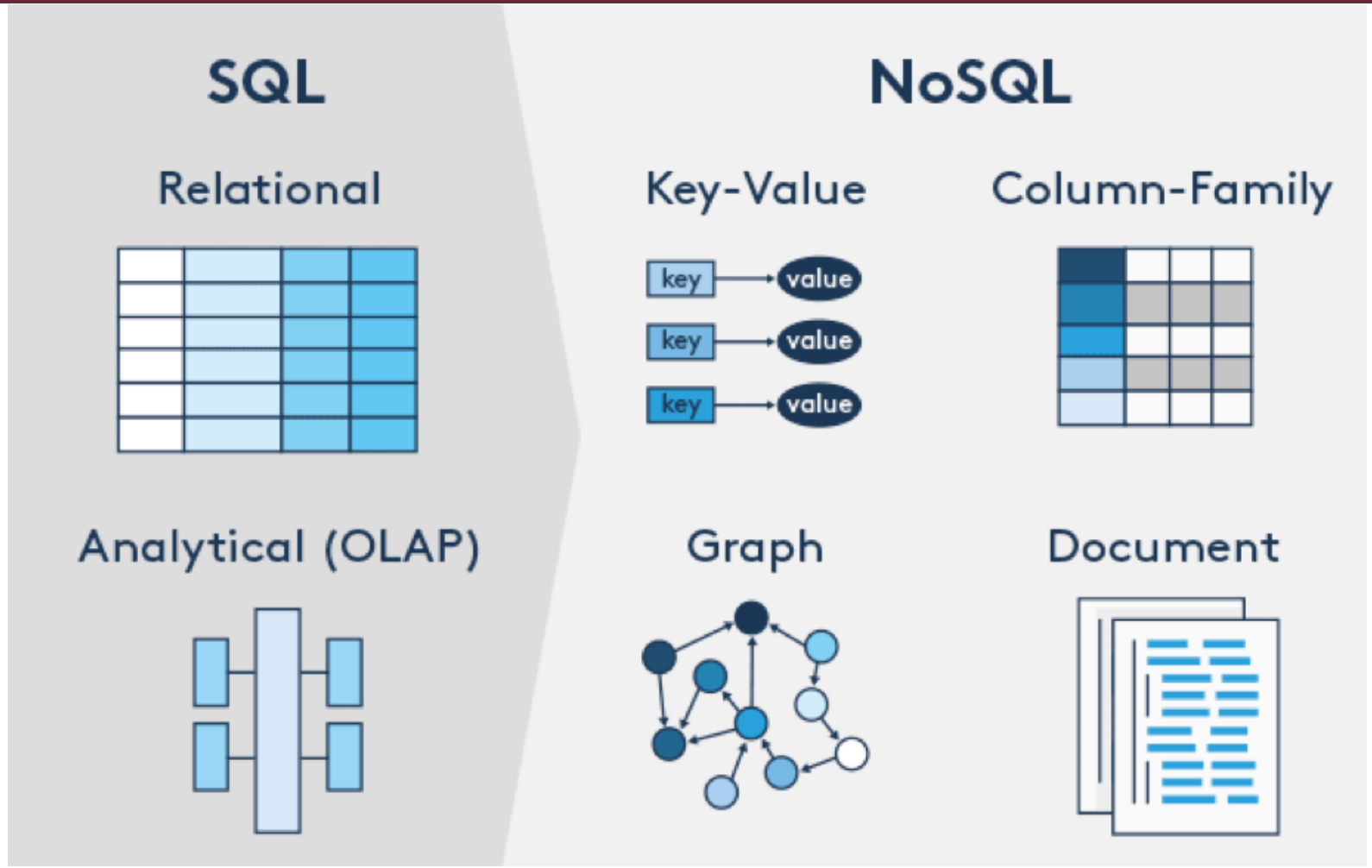
- Entender el funcionamiento de una base de datos no transaccional.
- Comprender el concepto de Big Data
- Estudiar los tipos de bases de datos: key-value, documentales, basadas en columnas y basadas en grafos.

INTRODUCCIÓN: EVOLUCIÓN DE LAS BASES DE DATOS

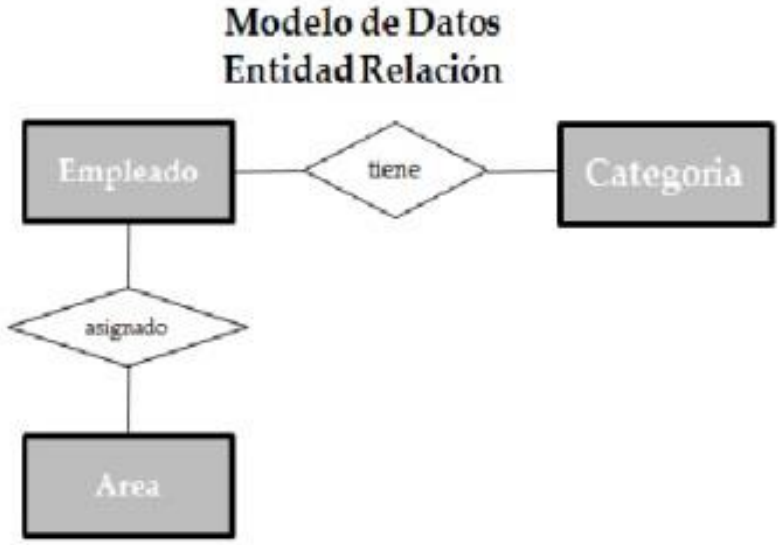


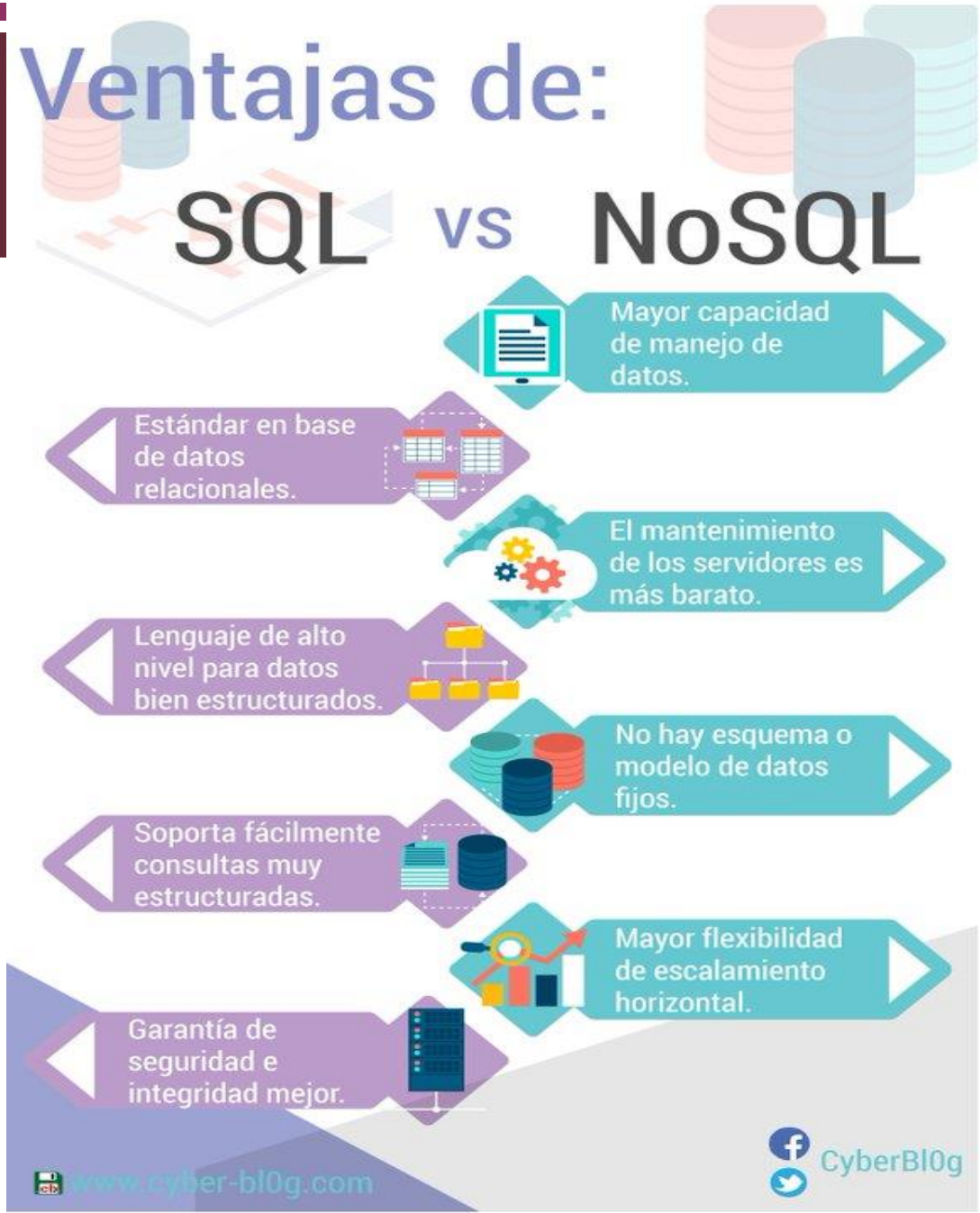

17

SQL VERUS NOSQL



SQL VERSUS OO VERSUS NOSQL





SQL VERSUS NOSQL

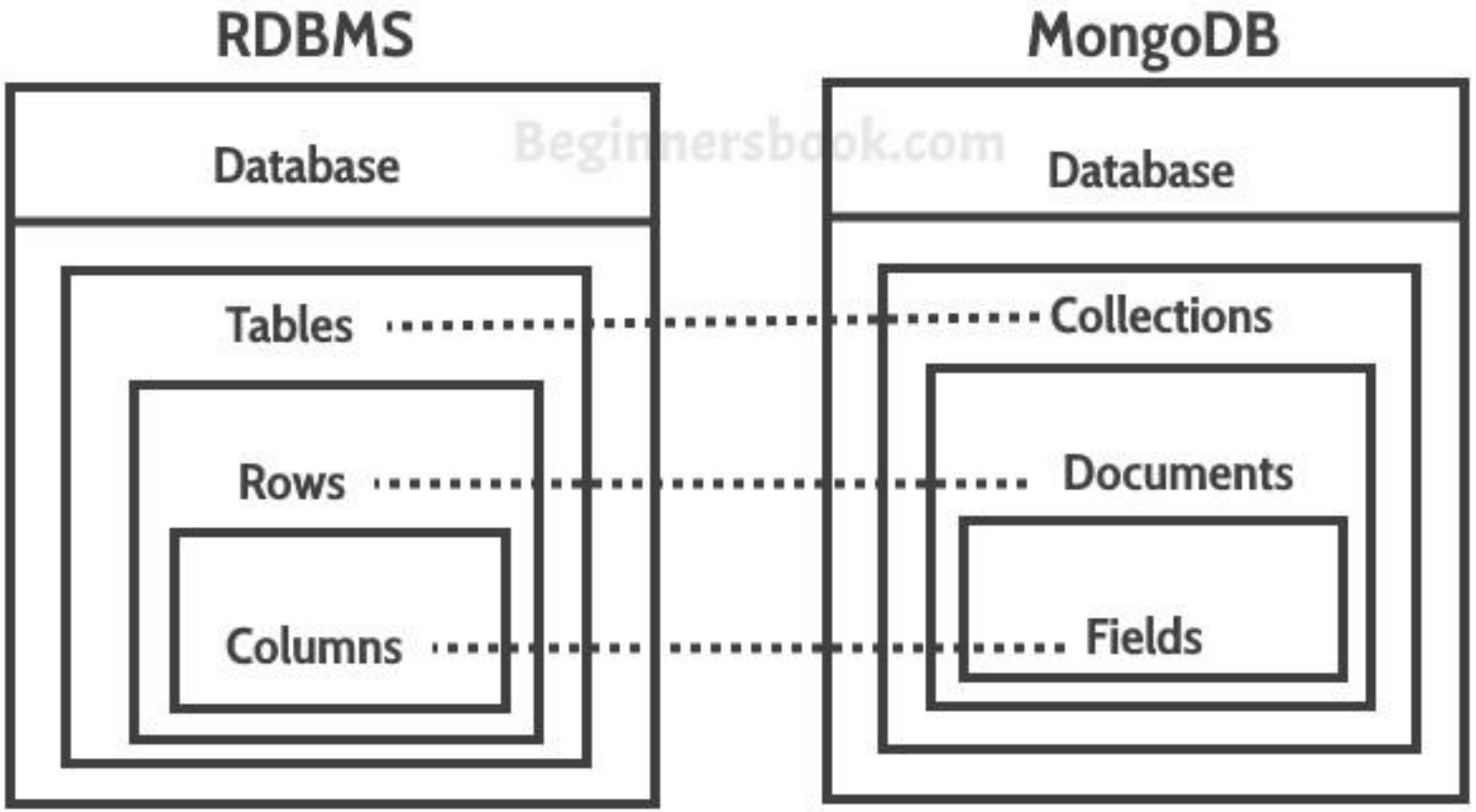
Relational data model

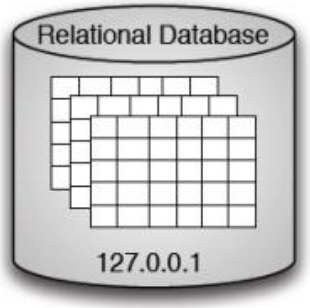
Highly-structured table organization with rigidly-defined data formats and record structure.

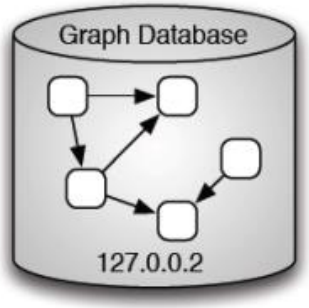
Document data model

Collection of complex documents with arbitrary, nested data formats and varying "record" format.

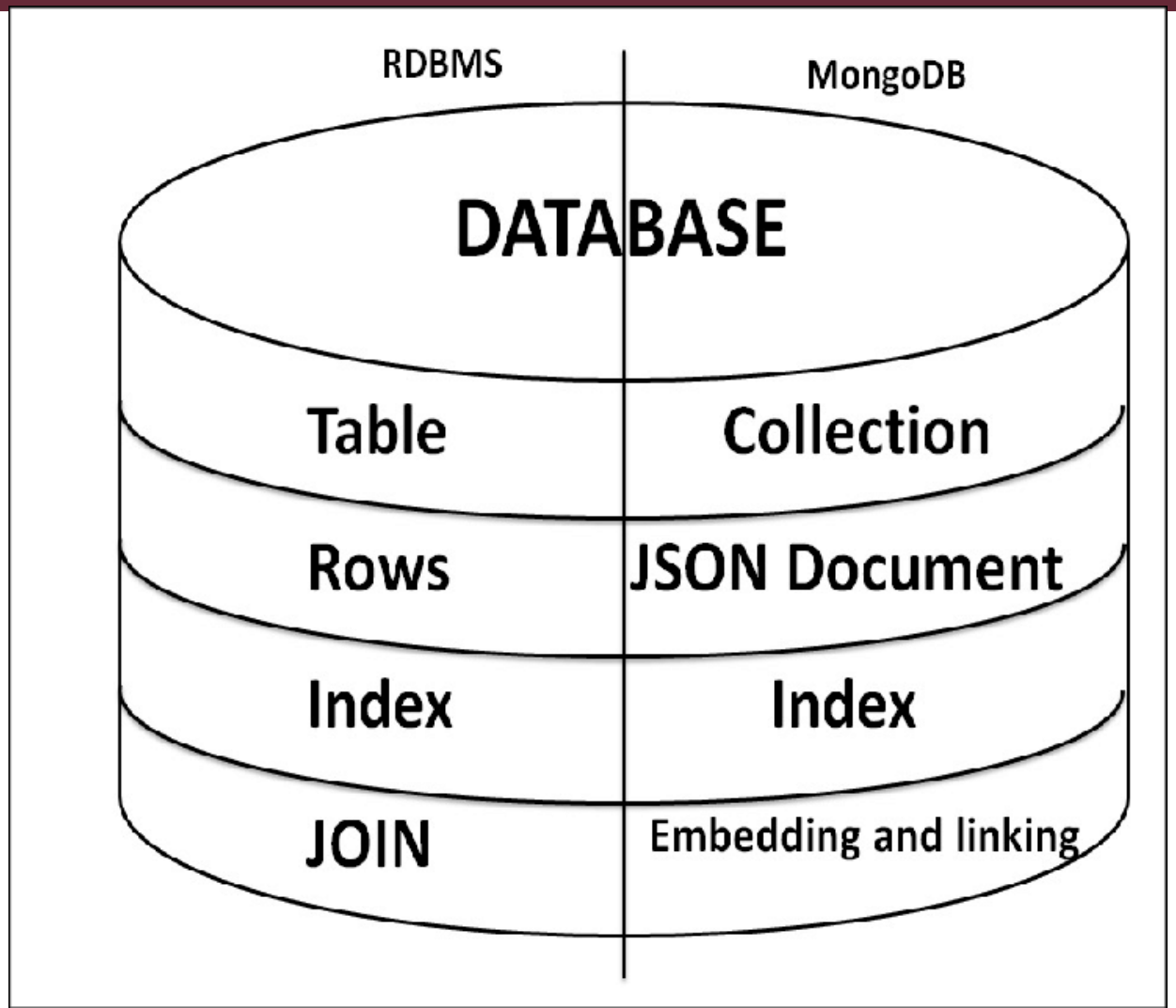
SQL VERSUS NOSQL



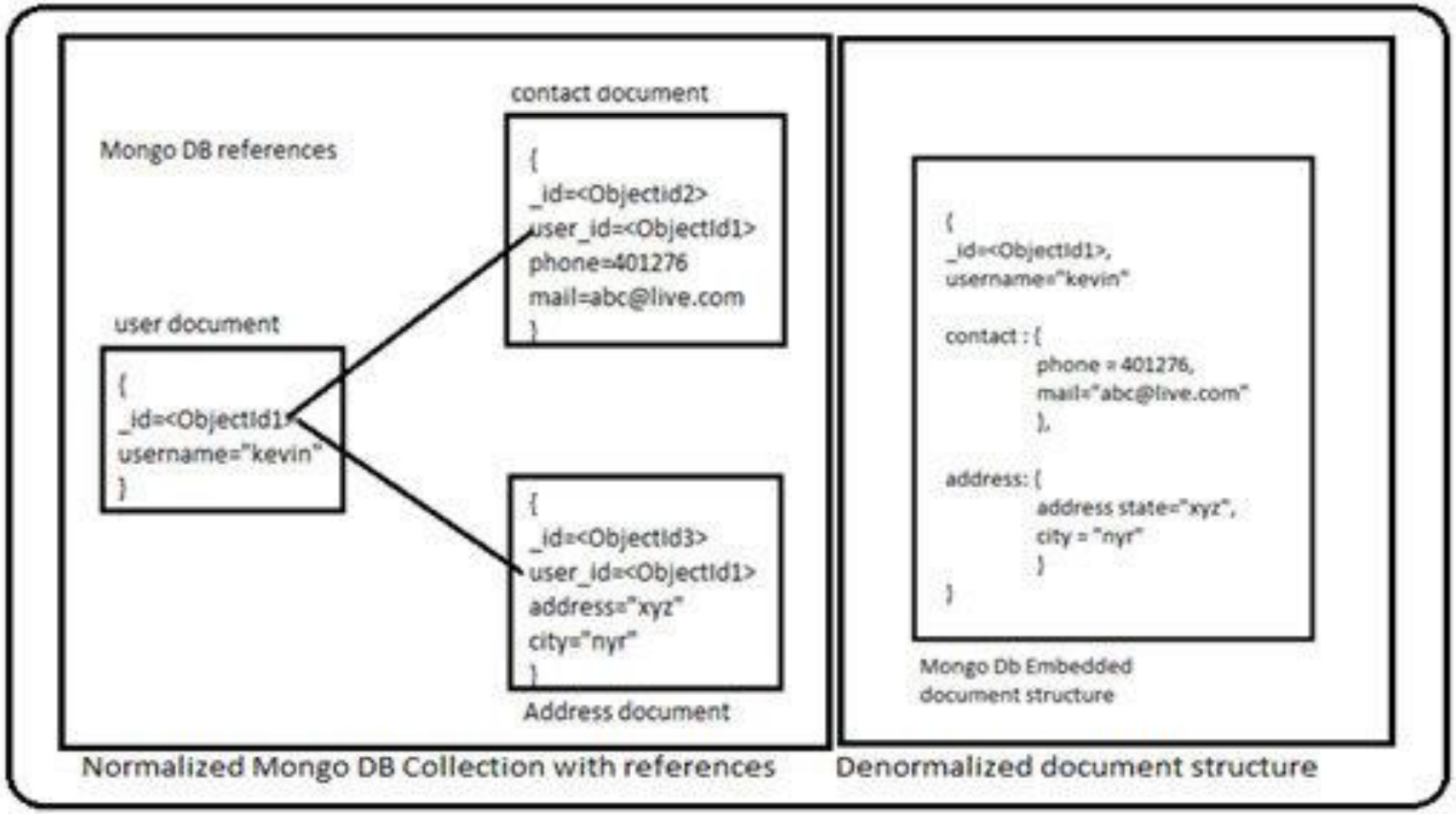




SQL VERSUS NOSQL



SQL VERSUS NOSQL



Mongo DB document structure data model

SQL VERSUS NOSQL

Característica	Relacional	No relacional
Rendimiento	Bajo	Alto
Disponibilidad	Bueno	Bueno
Consistencia	Bueno	Bajo
Confiabilidad	Bueno	Bajo
Almacenamiento de datos	Bueno para BBDD de mediano a gran tamaño	Optimizado para cantidades masivas de datos
Escalabilidad	Alto (más costoso)	Alto

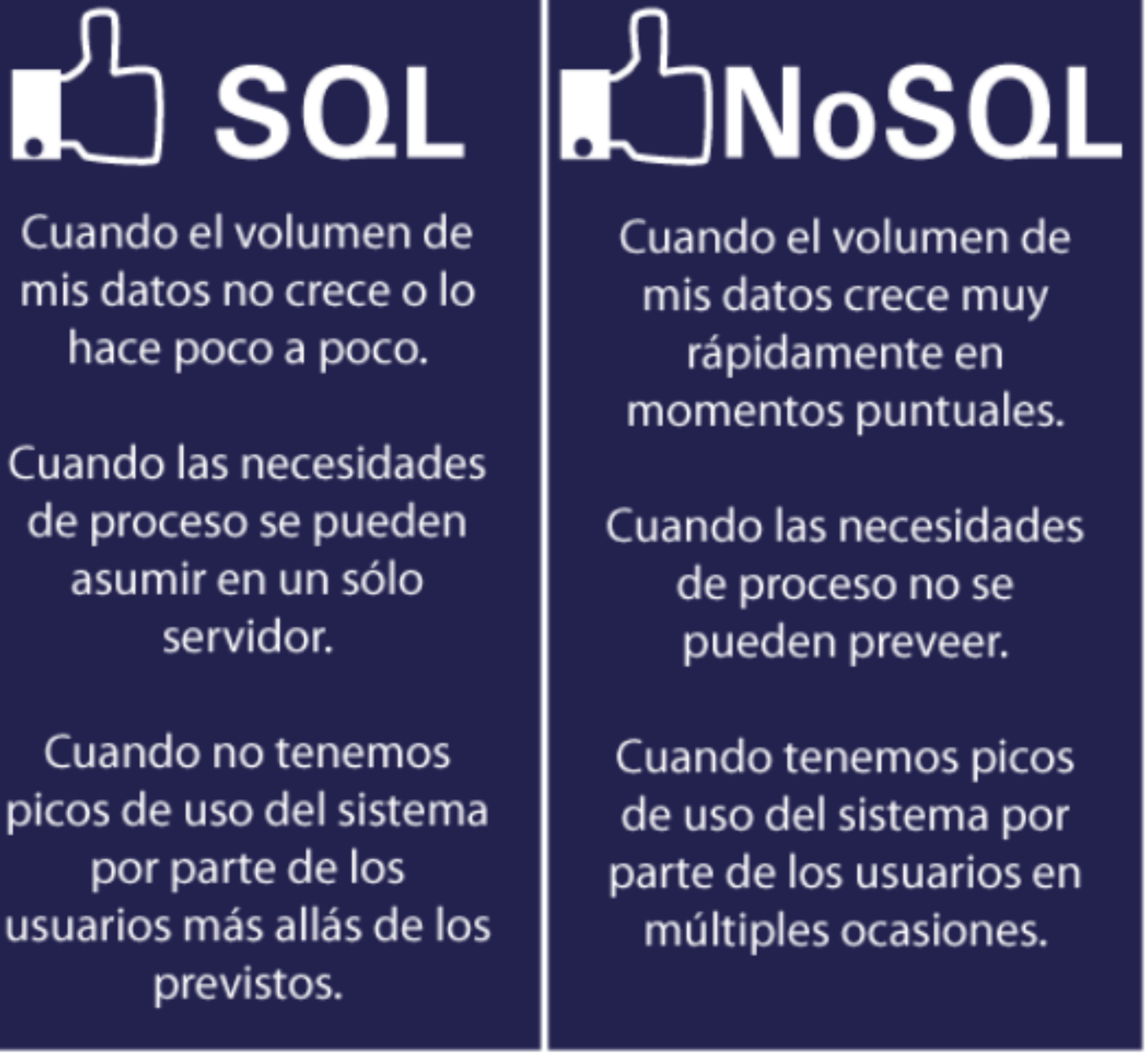
SQL VERSUS NOSQL

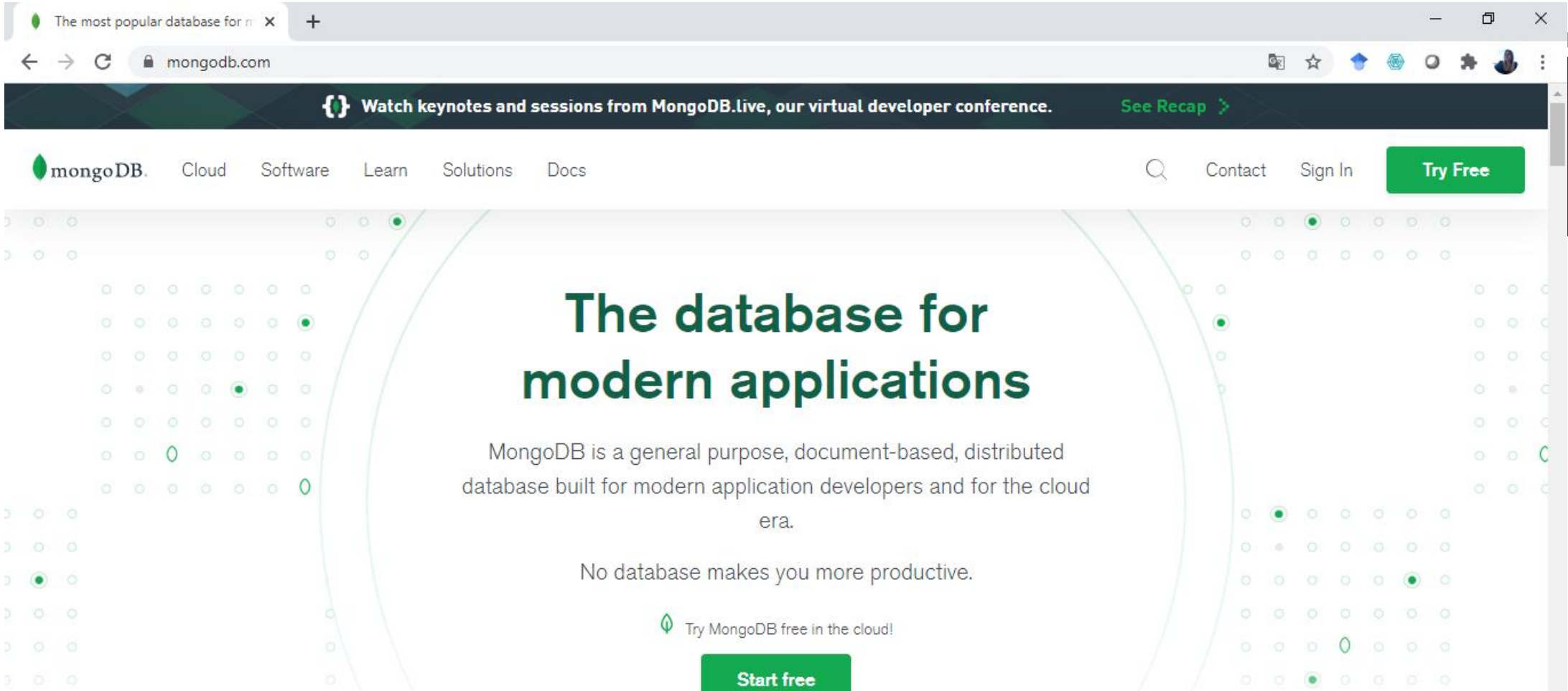
SQL	MongoDB
INSERT INTO nombreTabla **VALUES** (valorAtributo1, <, valorAtributo2...>);	db.nombreColeccion.insert({ atributo1: valorAtrib <, atributo2: valorAtributo2, ...>})
UPDATE nombreTabla **SET** atributo = nuevoValor <**WHERE** condición>	db.nombreColeccion.update({<condición>}, {$set: {atributo: nuevoValor} }<, [true|false],[true|false]>)
DELETE **FROM** nombreTabla <**WHERE** condición>	db.nombreColeccion.remove({<condición>})

MySQL	MongoDB
INSERT	
INSERT INTO account (`A/c number`, `first name`, `last name`) VALUES ('12345746352', 'Mark', 'Jacobs');	db.account.insert({ A/c number: "12345746352", first name: "Mark", last name: "Jacobs" });
UPDATE	
UPDATE account SET contact number = 9426227364 WHERE A/c number = '12345746352'	db.account.update({ A/c number: '12345746352' }, { $set: {contact number: 9426227364 } });
DELETE	
DELETE FROM account WHERE e-mail address = 'jv1994@gmail.com';	db.account.remove({ "E-mail address": "jv1994@gmail.com" });

student_id	age	score
1	12	77
2	12	68
3	11	75

```
[
 {
  "student_id":1,
  "age":12,
  "score":77
 },
 {
  "student_id":2,
  "age":12,
  "score":68
 },
 {
  "student_id":3,
  "age":11,
  "score":75
 }
]
```


SQL
Cuando el volumen de mis datos no crece o lo hace poco a poco.
Cuando las necesidades de proceso se pueden asumir en un sólo servidor.
Cuando no tenemos picos de uso del sistema por parte de los usuarios más allás de los previstos.
NoSQL
Cuando el volumen de mis datos crece muy rápidamente en momentos puntuales.
Cuando las necesidades de proceso no se pueden preveer.
Cuando tenemos picos de uso del sistema por parte de los usuarios en múltiples ocasiones.


The most popular database for
mongodb.com
Watch keynotes and sessions from MongoDB.live, our virtual developer conference.
See Recap
mongoDB.
Cloud
Software
Learn
Solutions
Docs
Contact
Sign In
Try Free
The database for modern applications
MongoDB is a general purpose, document-based, distributed database built for modern application developers and for the cloud era.
No database makes you more productive.
Try MongoDB free in the cloud!
Start free

mongoDB

MongoDB: Conceptos

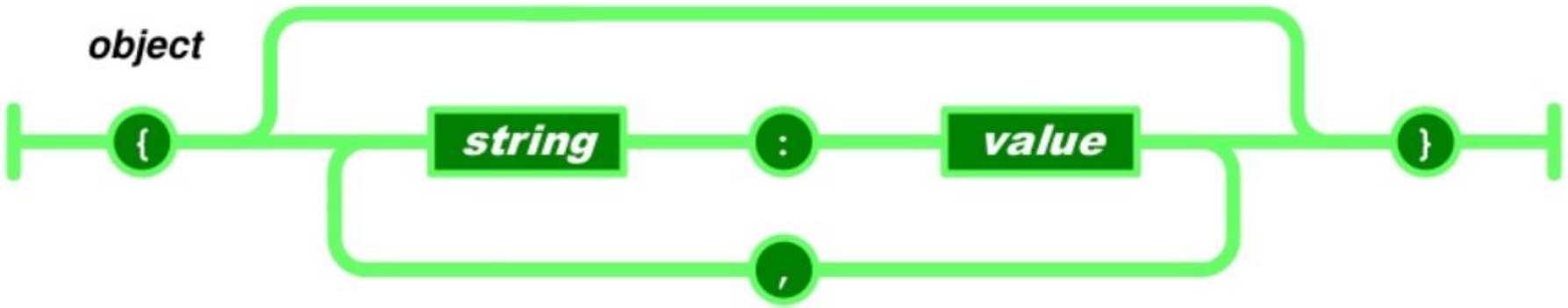
- Base de Datos multiplataforma, que provee alta performance, alta disponibilidad y escalamiento automático.

- Un registro en MongoDB es un **documento**, el cual es una estructura compuesta de pares **campo: valor**. Los documentos son similares a los objetos JSON.

https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

JSON: Conceptos

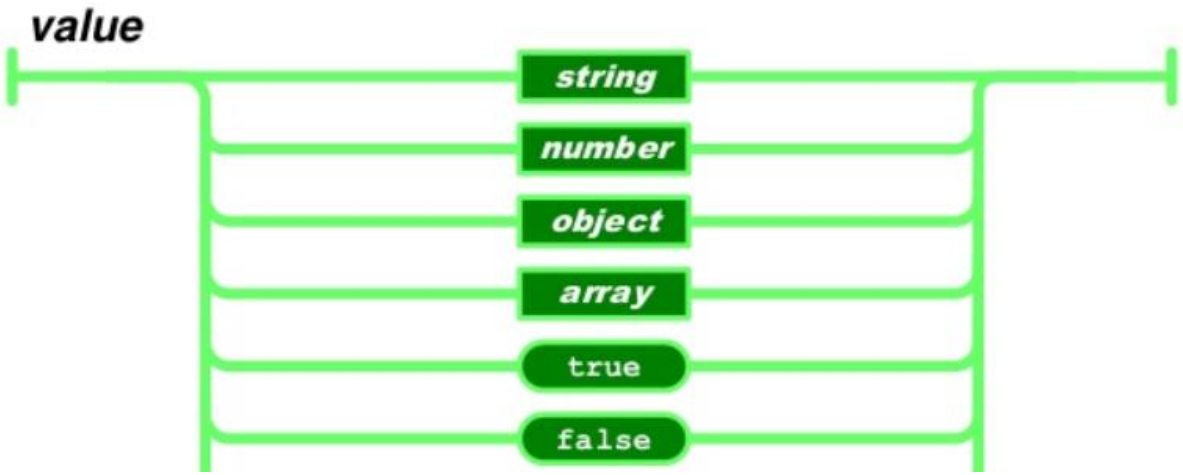
- JavaScript Object Notation, es un formato liviano para intercambio de datos.
- Es un formato de texto completamente independiente del lenguaje.
- Está constituido por dos estructuras:
  - Colección de pares de nombre / valor.
  - Una lista ordenada de valores.



https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

JSON: Values

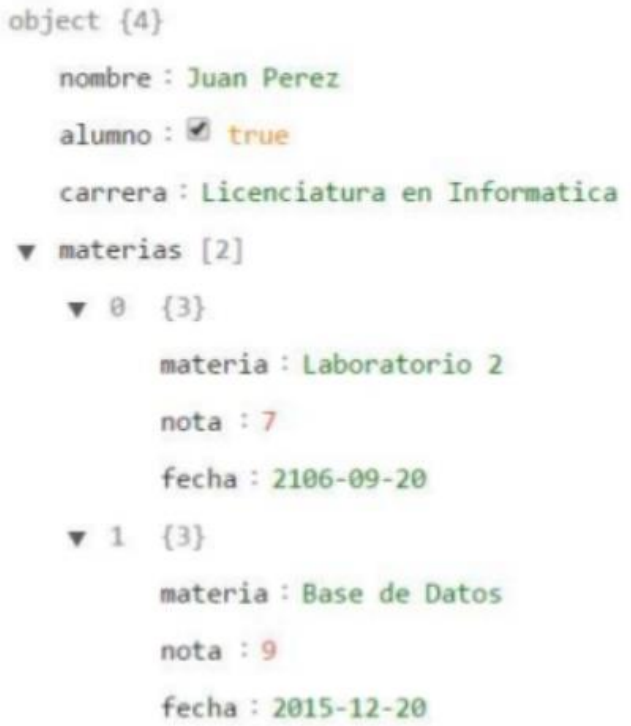
- Un valor puede ser una cadena de caracteres, un número, un boolean, un objeto o un array.



https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

JSON: Ejemplo

```
{       "nombre": "Juan Perez",
        "alumno": true,
        "carrera": "Licenciatura en Informatica",
        "materias":
        [
                {       "materia": "Laboratorio 2",
                        "nota": 7,
                        "fecha": "2106-09-20"
                },
                {       "materia": "Base de Datos",
                        "nota": 9,
                        "fecha": "2015-12-20"       }
        ]
}
```



https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

MongoDB: Documents

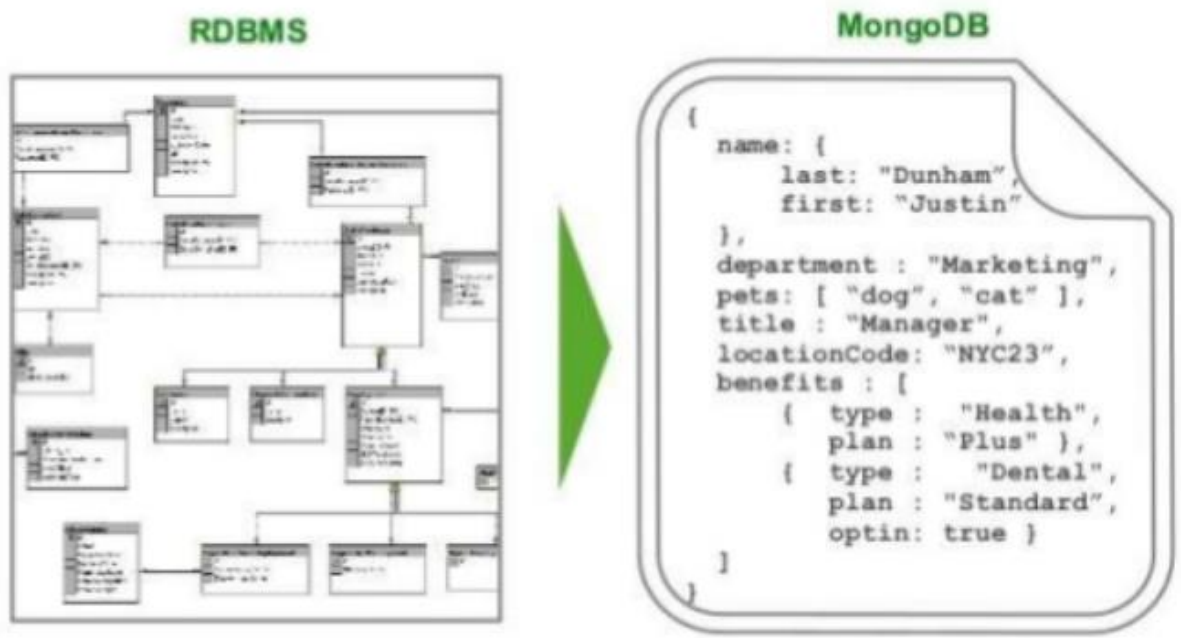
- MongoDB almacena los registros de datos como documentos BSON (Binary JSON).
- El valor de un campo puede ser cualquier tipo de dato BSON, incluyendo otro documento, array o un array de documentos.
- Cada documento incluye un ID para identificarlo univocamente.

```
{
   _id: ObjectId("5099803df3f4948bd2f98391"),
   name: { first: "Alan", last: "Turing" },
   birth: new Date('Jun 23, 1912'),
   death: new Date('Jun 07, 1954'),
   contribs: [ "Turing machine", "Turing test", "Turingery" ],
   views : NumberLong(1250000)
}
```


https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

MongoDB: Modelado

- Al momento de modelar un documento, se deben considerar la decisión de embeber los datos o utilizar referencias.



https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

MongoDB: Modelado

- Es posible embeber datos relacionados en una estructura simple o documento, estos esquemas son desnormalizados.
- Los modelos de datos embebidos se utilizan cuando:
  - Existe una relación “contiene” entre las entidades.
  - Existe una relación uno-a-muchos entre las entidades.

https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

MongoDB: Modelado

- Relaciones **uno-a-uno** entre entidades, es posible embeber directamente los datos en el documento.
  - En el ejemplo, el domicilio del cliente está embebido en la entidad.

```
{
    _id: "joe",
    name: "Joe Bookreader",
    address: {
                street: "123 Fake Street",
                city: "Faketon",
                state: "MA",
                zip: "12345"
              }
}
```

https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

MongoDB: Modelado

- Es posible resolver relaciones **uno-a-muchos** embebiendo la información en el documento.

```
{
    _id: "joe",
    name: "Joe Bookreader",
    addresses: [
                {
                  street: "123 Fake Street",
                  city: "Faketon",
                  state: "MA",
                  zip: "12345"
                },
                {
                  street: "1 Some Other Street",
                  city: "Boston",
                  state: "MA",
                  zip: "12345"
                }
              ]
}
```

https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

MongoDB: Modelado

- Es posible (para evitar la redundancia), resolver las relaciones **uno-a-muchos** utilizando referencias.

```
{
   title: "MongoDB: The Definitive Guide",
   author: [ "Kristina Chodorow", "Mike Dirolf" ],
   published_date: ISODate("2010-09-24"),
   pages: 216,
   language: "English",
   publisher: {
              name: "O'Reilly Media",
              founded: 1980,
              location: "CA"
            }
}
```

```
{
   name: "O'Reilly Media",
   founded: 1980,
   location: "CA",
   books: [12346789, 234567890, ...]
}

{
    _id: 123456789,
    title: "MongoDB: The Definitive Guide",
    author: [ "Kristina Chodorow", "Mike Dirolf" ],
    published_date: ISODate("2010-09-24"),
    pages: 216,
    language: "English"
}
```

https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

MongoDB: CRUD

CREATE READ UPDATE DELETE

C R U D

- **Create Operation:**
  - db.collection.insert()
  - db.collection.insertOne()
  - db.collection.insertMany()
- **Read Operation:**
  - db.collection.find()
- **Update Operation:**
  - db.collection.update()
  - db.collection.updateOne()
  - db.collection.updateMany()
  - db.collection.replaceOne()
- **Delete Operation:**
  - db.collection.remove()
  - db.collection.deleteOne()
  - db.collection.deleteMany()

https://es.slideshare.net/jazcurra/nosql-mongodb-67920192

MySQL	MongoDB

`INSERT INTO users (user_id, age, status)`
`VALUES ('bcd001', 45, 'A')`
	`db.users.insert({`
	`  user_id: 'bcd001',`
	`  age: 45,`
	`  status: 'A'`
	`})`

`SELECT * FROM users`	`db.users.find()`

`UPDATE users SET status = 'C'`
`WHERE age > 25`
	`db.users.update(`
	`  { age: { $gt: 25 } },`
	`  { $set: { status: 'C' } },`
	`  { multi: true }`
	`)`

¿Consultas?

UAI
Universidad Abierta Interamericana